

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«Северный (Арктический) федеральный университет имени М.В. Ломоносова»

Высшая школа информационных технологий и автоматизированных систем

ЛАБОРАТОРНАЯ РАБОТА №2

По дисциплине: Защита информации в системах управления базами данных

На тему Системы хранения данных. Часть 2

Выполнил обучающийся:

Грозов Илья Владимирович

Направление подготовки / специальность:

10.03.01 Информационная безопасность

Курс: 3

Группа: 151113

Руководитель: Зубарев Александр Андреевич, ст.

преподаватель

Отметка о зачете

Руководитель

А.А. Зубарев.

Архангельск 2023

ЗАДАНИЕ

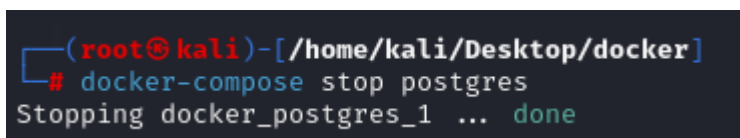
Получить практический навык PostgreSQL CIS Benchmarks и ПРД

ХОД РАБОТЫ

1. ПОДГОТОВКА КОНТЕЙНЕРА DOCKER-COMPOSE

1.1 Остановка контейнера docker-compose.yml

Чтобы изменить метод аутентификации на изначальный требуется остановить контейнер docker при помощи команды `docker-compose stop postgres`. Остановка контейнера отображена на рисунке 1.1.1

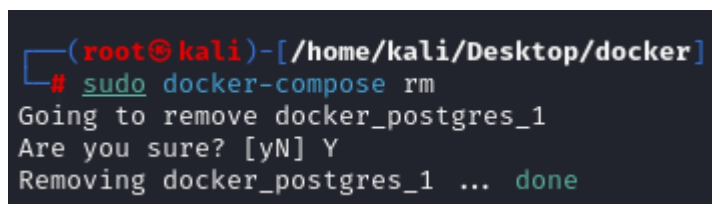


```
(root@kali)-[/home/kali/Desktop/docker]
# docker-compose stop postgres
Stopping docker_postgres_1 ... done
```

Рисунок 1.1.1 – Остановка контейнера

1.2 Удаление контейнера docker-compose

Необходимо удалить контейнер docker-compose. Удаление контейнера docker-compose производится с помощью команды: `sudo docker-compose rm`. Удаление контейнера необходимо подтвердить. Удаление контейнера отображено на рисунке 1.2.1.



```
(root@kali)-[/home/kali/Desktop/docker]
# sudo docker-compose rm
Going to remove docker_postgres_1
Are you sure? [yN] Y
Removing docker_postgres_1 ... done
```

Рисунок 1.2.1 – Удаление контейнера

1.3 Возвращение исходной конфигурации docker-compose.yml

Вернем файл `docker-compose.yml` в изначальное состояние. Уберем метод аутентификации. Исходный файл `docker-compose.yml` с убраным методом аутентификации предоставлен на рисунке 1.3.1

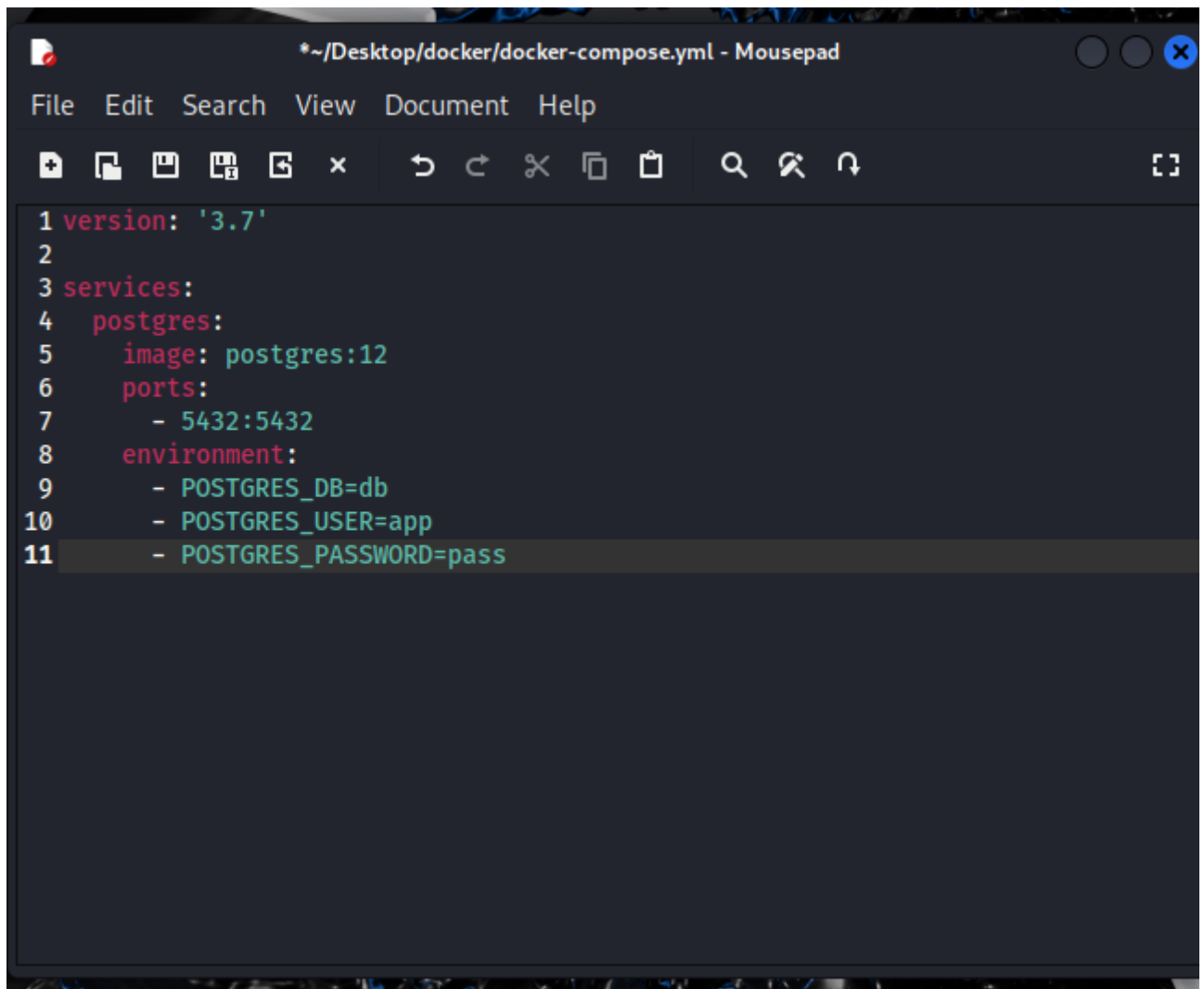


Рисунок 1.3.1 - Исходный файл docker-compose.yml

1.4 Запуск контейнера

Запустим контейнер docker по исходному файлу docker-compose.yml при помощи команды: `docker-compose up`. Запуск контейнера docker приведен на рисунке 1.4.1

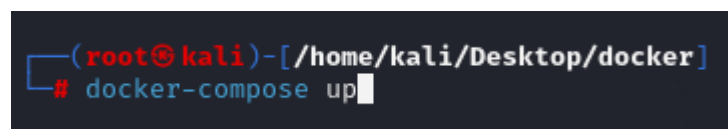


Рисунок 1.4.1 – Запуск контейнера docker

Итог запуска контейнера docker приведен на рисунке 1.4.2.

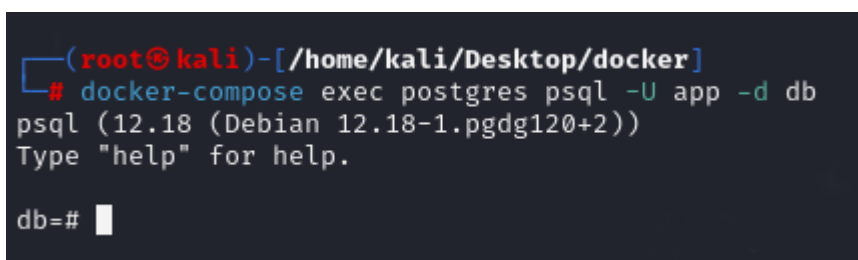
```
postgres_1 |
postgres_1 | /usr/local/bin/docker-entrypoint.sh: ignoring /docker-entrypoint-initdb.d/*
postgres_1 |
postgres_1 | waiting for server to shut down... 2024-03-10 08:44:31.392 UTC [49] LOG:  received fast shutdown request
postgres_1 | 2024-03-10 08:44:31.396 UTC [49] LOG:  aborting any active transactions
postgres_1 | 2024-03-10 08:44:31.398 UTC [51] LOG:  background worker "logical replication launcher" (PID 56) exited with exit code 1
postgres_1 | 2024-03-10 08:44:31.398 UTC [51] LOG:  shutting down
postgres_1 | 2024-03-10 08:44:31.425 UTC [49] LOG:  database system is shut down
postgres_1 | done
postgres_1 | server stopped
postgres_1 |
postgres_1 | PostgreSQL init process complete; ready for start up.
postgres_1 |
postgres_1 | 2024-03-10 08:44:31.515 UTC [1] LOG:  starting PostgreSQL 12.18 (Debian 12.18-1.pgdg120+2) on x86_64-pc-linux-gnu, compiled by gcc (Debian 12.2.0-14) 12.2.0, 64-bit
postgres_1 | 2024-03-10 08:44:31.515 UTC [1] LOG:  listening on IPv4 address "0.0.0.0", port 5432
postgres_1 | 2024-03-10 08:44:31.515 UTC [1] LOG:  listening on IPv6 address "::", port 5432
postgres_1 | 2024-03-10 08:44:31.523 UTC [1] LOG:  listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
postgres_1 | 2024-03-10 08:44:31.569 UTC [78] LOG:  database system was shut down at 2024-03-10 08:44:31 UTC
postgres_1 | 2024-03-10 08:44:31.578 UTC [1] LOG:  database system is ready to accept connections
```

Рисунок 1.4.2 – Итог запуска контейнера docker

2. БАЗА ДАННЫХ SQL

2.1 Подключение к базе данных

Выполним подключение к базе данных при помощи команды: `docker-compose exec postgres psql -U app -d db`. Подключение к базе данных отображено на рисунке 2.1.1.



```
(root@kali)-[/home/kali/Desktop/docker]
# docker-compose exec postgres psql -U app -d db
psql (12.18 (Debian 12.18-1.pgdg120+2))
Type "help" for help.

db=#
```

Рисунок 2.1.1 - Подключение к базе данных

3. ПОДГОТОВКА ДАННЫХ. РАБОТА С ОБОЛОЧКОЙ SQL

3.1 Отображение существующих системных ролей базы данных

Для отображения существующих системных ролей воспользуемся командой: `SELECT * FROM pg_roles`. Отображение существующих системных ролей приведено на рисунке 3.1.1

```
db=# SELECT * FROM pg_roles;
```

rolname	rolsuper	rolinherit	rolcreatorole	rolcreatedb	rolcanlogin	rolreplication	rolconnlimit	rolpassword	rolvaliduntil	rolbypassrls	rolconfig	oid
pg_signal_backend	f	t	f	f	f	f	-1	*****		f		4200
pg_read_server_files	f	t	f	f	f	f	-1	*****		f		4569
app	t	t	t	t	t	t	-1	*****		t		10
pg_write_server_files	f	t	f	f	f	f	-1	*****		f		4570
pg_execute_server_program	f	t	f	f	f	f	-1	*****		f		4571
pg_read_all_stats	f	t	f	f	f	f	-1	*****		f		3375
pg_monitor	f	t	f	f	f	f	-1	*****		f		3373
pg_read_all_settings	f	t	f	f	f	f	-1	*****		f		3374
pg_stat_scan_tables	f	t	f	f	f	f	-1	*****		f		3377

```
(9 rows)
```

db=#

Рисунок 3.1.1 – Отображение системных ролей

3.2 Создание новой роли reader с правом входа

Создадим новую роль reader с правом входа при помощи команды: `CREATE ROLE reader LOGIN PASSWORD 'secret'`. Создание новой роли отображено на рисунке 3.2.1

```
db=# CREATE ROLE reader LOGIN PASSWORD 'secret';
CREATE ROLE
db=#
```

Рисунок 3.2.1 – Создание новой роли reader с правом входа

3.3 Проверка существующих баз данных

Проверим существующие базы данных в системе при помощи команды: `SELECT * FROM pg_database`. Существующие базы данных с их владельцами отображены на рисунке 3.3.1

```
db=# SELECT * FROM pg_database;
```

oid	datname	datdba	encoding	datcollate	datctype	datistemplate	dataallowconn	datconnlimit	datlastsysoid	datfrozenxid	datminmxid	dattablespace	datacl
13481	postgres	10	6	en_US.utf8	en_US.utf8	f	t	-1	13480	480	1	1663	
16384	db	10	6	en_US.utf8	en_US.utf8	f	t	-1	13480	480	1	1663	
1	template1	10	6	en_US.utf8	en_US.utf8	t	t	-1	13480	480	1	1663	{-c/app,app-CTC/app}
13480	template0	10	6	en_US.utf8	en_US.utf8	t	f	-1	13480	480	1	1663	{-c/app,app-CTC/app}

```
(4 rows)
```

Рисунок 3.3.1 – Существующие базы данных с их владельцами

3.4 Создание тестовой базы данных

Создадим тестовую базу данных при помощи команды: `test CREATE DATABASE test`. Создание тестовой базы данных отображено на рисунке 3.4.1.

```
db=# CREATE DATABASE test;  
CREATE DATABASE  
db=#
```

Рисунок 3.4.1 – Создание тестовой базы данных

3.5 Подключение к созданной базе данных

Выполним подключение к тестовой базе данных при помощи команды: `\c test`. Подключение к тестовой базе данных отображено на рисунке 3.5.1.

```
db=# \c test  
You are now connected to database "test" as user "app".  
test=#
```

Рисунок 3.5.1 – Подключение к тестовой базе данных

3.6 Создание таблицы records в базе данных

Создадим таблицу records в базе данных при помощи команды: `CREATE TABLE records (value TEXT, status TEXT, created TIMESTAMP DEFAULT CURRENT_TIMESTAMP)`. Создание таблицы отображено на рисунке 3.6.1

```
test=# CREATE TABLE records (value TEXT, status TEXT, created TIMESTAMP DEFAULT CURRENT_TIMESTAMP);  
CREATE TABLE  
test=#
```

Рисунок 3.6.1 – Создание таблицы records

3.7 Добавление записи в таблицу

В созданную таблицу records добавим запись содержания: `INSERT INTO records(value, status) VALUES ('transfer money from 55** **** 0001 to 42** **** 0002', 'success')`. Добавление записи отображено на рисунке 3.7.1

```
test=# INSERT INTO records(value, status) VALUES ('transfer money from 55** **** 0001 to 42** **** 0002', 'success');  
INSERT 0 1  
test=#
```

Рисунок 3.7.1 – Добавление записи в таблицу

3.8 Предоставление прав ролям базы данных на чтение содержимого

Предоставим права для роли reader на выполнение чтения содержимого таблицы records при помощи команды: GRANT SELECT ON records TO reader. Добавление прав на чтение таблицы отображено на рисунке 3.8.1

```
test=# GRANT SELECT ON records TO reader
test=#
```

Рисунок 3.8.1 – Добавление прав на чтение таблицы

3.9 Просмотр предоставленных прав

Выполним просмотр предоставленных прав для роли при помощи команды: \dp. Предоставленные права отображены на рисунке 3.9.1

```
test=# \dp
      Access privileges
 Schema | Name   | Type | Access privileges | Column privileges | Policies
-----+-----+-----+-----+-----+-----
 public | records | table |                    |                   |
(1 row)

test=#
```

Рисунок 3.9.1 – Просмотр предоставленных прав

4. ПРОВЕРКА ПРД

4.1 Получение оболочки для пользователя

Выполним получение оболочки psql для пользователя reader при помощи команды: `docker-compose exec postgres psql -U reader -d test`, предварительно произведя выход из базы данных командой `\q`. Выход из базы данных предоставлен на рисунке 4.1.1. Получение оболочки для пользователя предоставлено на рисунке 4.1.2

```
test-# exit
Use \q to quit.
test-# \q

(root@kali)-[/home/kali/Desktop/docker]
#
```

Рисунок 4.1.1 – Выход из базы данных

```
(root@kali)-[/home/kali/Desktop/docker]
# docker-compose exec postgres psql -U reader -d test
psql (12.18 (Debian 12.18-1.pgdg120+2))
Type "help" for help.

test=>
```

Рисунок 4.1.2 – Получение оболочки для пользователя

4.2 Выполнение запроса на чтение

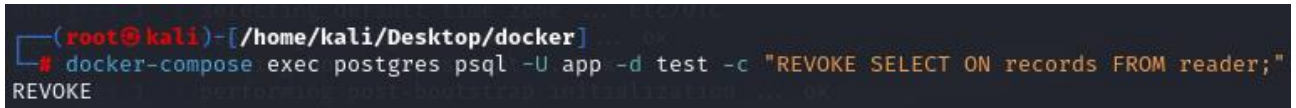
Выполним запрос на чтение при помощи команды: `docker-compose exec postgres psql -U app -d test -c «GRANT SELECT ON records TO reader;»`. Выполнение запроса на чтение предоставлено на рисунке 4.2.1

```
(root@kali)-[/home/kali/Desktop/docker]
# docker-compose exec postgres psql -U app -d test -c "GRANT SELECT ON records TO reader;"
GRANT
SELECT
SELECTING DEFAULT SHARED BUFFERS ... 128MB
SELECTING DEFAULT TIME ZONE ... Etc/UTC
ok
running postgresql script ... ok
```

Рисунок 4.2.1 – Выполнение запроса на чтение

4.3 Отмена прав ролей на чтение

Отменим права на чтение у роли reader при помощи команды: `docker-compose exec postgres psql -U app -d test -c «REVOKE SELECT ON records FROM reader;»`. Отмена прав на чтение у роли reader предоставлена на рисунке 4.3.1.

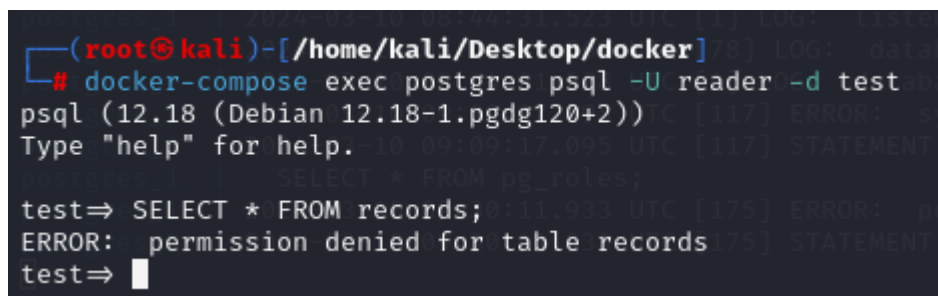


```
(root@kali)-[/home/kali/Desktop/docker]
# docker-compose exec postgres psql -U app -d test -c "REVOKE SELECT ON records FROM reader;"
REVOKE
```

Рисунок 4.3.1 – Отмена прав на чтение

4.4 Выполнение запроса на чтение

Выполним запрос на чтение при помощи команды: `SELECT * FROM records`, предварительно произведя обратный вход в базу данных. Ответ в запрос на чтение предоставлен на рисунке 4.4.1



```
(root@kali)-[/home/kali/Desktop/docker]
# docker-compose exec postgres psql -U reader -d test
psql (12.18 (Debian 12.18-1.pgdg120+2))
Type "help" for help.

test=> SELECT * FROM records;
ERROR: permission denied for table records
test=> █
```

Рисунок 4.4.1 – Ответ в запрос на чтение

База данных сообщает нам об ошибке – отсутствуют права доступа. У пользователя от имени которого выполняется запрос отсутствуют права на доступ к таблице «records»

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие методы не рекомендуется использовать для удалённых подключений?

Для удаленных подключений к базе данных не рекомендуется использовать незащищенные методы, такие как открытые соединения или передача данных через незашифрованные протоколы, не рекомендуется использовать стандартные порты для удаленного доступа к базе данных

2. Какие методы рекомендуется использовать для удалённых подключений?

Для удаленных подключений к базе данных рекомендуется использовать защищенные методы, такие как VPN или SSH, следует удостовериться, что база данных настроена на безопасный режим работы с удаленными подключениями, например, путем ограничения доступа только к определенным IP-адресам

ВЫВОД

В ходе выполнения лабораторной работы по теме: «Работа со средствами дизассемблирования и отладки» получили практический навык PostgreSQL CIS Benchmarks и ПРД